IN THE CLAIMS

Please amend the claims as follows:

1-27. Cancelled

28. (New) A computer system comprising:

a source file repository storing a plurality of active source files belonging to a component; and

a central compilation service communicatively coupled with the source file repository that, upon receiving an activation request for at least one inactive source file of the component, compiles the component using the at least one inactive source file and, in case the compilation is successfully completed, initiates a transfer of the at least one inactive source file to the plurality of active source files.

29. (New) The computer system of claim 28, wherein the transfer comprises adding the at least one inactive source file to the plurality of active source files if the inactive source file has no corresponding active source file.

30. (New) The computer system of claim 28, wherein the transfer of the at least one inactive source file is in response to replacing at least one corresponding active source file with the at least one inactive source file in case the corresponding active source file is outdated.

31. (New) The computer system of claim 28, wherein the transfer comprises replacing at least one corresponding active source file with the at least one inactive source file if the active source file is outdated.

32. (New) The computer system of claim 28, further comprising a runtime archive storage to store a compilation result of the component in case the compilation of the component is successfully completed.

33. (New) The computer system of claim 28, wherein the at least one inactive source file is stored in a further source file repository.

34. (New) The computer system of claim 28, wherein the central compilation service notifies a changer of the inactive source file in case the compilation of the component fails.

35. (New) The computer system of claim 28, wherein the central compilation service assigns a component status to the component depending on the result of the compilation, the component status being ready in case the compilation of the component is successfully completed.

36. (New) The computer system of claim 28, wherein the central compilation service assigns a component status to the component depending on the result of the compilation, the component status being broken in case the compilation of the component fails.

37. (New) The computer system of claim 28, wherein the central compilation service assigns a component status of dirty to a further component that depends on the component in case the compilation of the component is successfully completed.

38. (New) The computer system of claim 28, wherein the central compilation service performs an incremental build when compiling the component if the component has a dependency on a further component, the incremental build using a component dependency evaluator to determine the dependency and to provide a previously obtained compilation result of the further component, and to provide the at least one inactive source file and the plurality of active source files of the component.

39. (New) The computer system of claim 28, wherein the central compilation service performs a parallel build when compiling the component by evaluating dependencies of the component on further components and compiling the component and at least one of the further components in parallel based on the dependencies.

40. (New) The computer system of claim 39, wherein the parallel build is performed by a cluster of build computers.

41. (New) An article of manufacture comprising instructions that when loaded into a memory of a computer system and executed by at least one processor of the computer system, cause the at least one processor to:

receive an activation request for at least one inactive source file of the component;

compile the component using the at least one inactive source file; and

initiate a transfer of the at least one inactive source file to a plurality of active source files, in case the compilation is successfully completed.

42. (New) The article of claim 41 further comprising instructions that, when loaded into the memory of the computer system and executed by the at least one processor of the computer system, cause the at least on processor to notify a changer of the inactive source file, in case the compilation fails.

43. (New) The article of claim 41, wherein the instructions that cause the at least one processor to compile the component comprise instructions that, when loaded into the memory of the computer system and executed by the one or more processors cause the at least one processor to perform the compilation as an incremental build, the component having a dependency on a further component, the incremental build using a component dependency evaluator to determine the dependency and to provide a previously obtained compilation result of the further component, and to provide the at least one inactive source file and the plurality active source files of the component.

44. (New) The article of claim 41, wherein the instructions that cause the at least one processor to compile the component comprise instructions that, when loaded into the memory of the computer system and executed by the one or more processors cause the at least one processor to perform the compilation as a parallel build by evaluating dependencies of the component on further components and compiling the component and at least one of the further components in parallel based on the dependencies.

45. (New) The article of claim 44, wherein the parallel build is performed by a cluster of build computers.

46. (New) A source file activation method comprising:

receiving at a central compilation service an activation request for at least one inactive source file of a component;

compiling the component using the at least one inactive source file; and

initiating a transfer of the at least one inactive source file to a plurality of active source files, in case the compilation is successfully completed.

47. (New) The method of claim 46 further comprising notifying a changer of the inactive source file, in case the compilation fails.

48. (New) The method of claim 46, wherein the transfer of the at least one inactive source file comprises adding the at least one inactive source file to the plurality of active source files in case the plurality has no corresponding active source file.

49. (New) The method of claim 46, wherein the transfer of the at least one inactive source file comprises replacing at least one corresponding active source file with the at least one inactive source file in case the corresponding active source file is outdated

50. (New) The method of claim 46, further comprising storing a compilation result of the component in a runtime archive storage in case the compilation of the component is successfully completed.

51. (New) The method of claim 50, further comprising storing a compilation result of the component in a runtime archive storage in case the compilation of the component is successfully completed.

52. (New) The method of claim 46, further comprising:

assigning a component status to the component depending on the result of the compilation, the component status being selected from the group of: ready in case the compilation of the component is successfully completed, and broken in case the compilation of the component fails; and

assigning a component status of dirty to a further component in case the further component depends on the component and the compilation of the component is successfully completed.

53. (New) The method of claim 46, wherein the compilation is performed as an incremental build, the component having a dependency on a further component, the incremental build using a component dependency evaluator to determine the dependency and to provide a previously obtained compilation result of the further component, and to provide the at least one inactive source file and the plurality active source files of the component.

54. (New) The method of 46, wherein the compilation is performed as a parallel build by evaluating dependencies of the component on further components and compiling the component and at least one of the further components in parallel based on the dependencies.

55. (New) A method for validating software comprising:

retrieving a source file of a component referencing a referenced component from a source file repository of a source control system with a local file system;

obtaining a compilation result of the referenced component from a runtime archive storage with the local file system;

receiving a modification of the source file with an integrated development environment;

transferring the source file to a local build tool within the integrated development environment upon having received the modification;

retrieving the compilation result from the local file system with the local build tool;

locally compiling the component that includes the modified source file by using the compilation result of the referenced component resulting in a new compilation result of the component with the local build tool;

storing the new compilation result in the local file system;

checking in the modified source file into the source file repository, the modified source file becoming an inactive source file of the source file repository;

launching an activation request with regards to the inactive source file directed to a central compilation service; and

loading the inactive source file and corresponding active source files of the component from the source file repository with the central compilation service.

56. (New) The method of claim 55 further comprising:

retrieving the compilation result of the referenced component from the runtime archive storage;

centrally compiling the component with the central compilation service; and

triggering the activation of the successfully compiled inactive source file in the source file repository in case of successful central compilation.

57. (New) The method of claim 56 further comprising the integrated development environment deploying the new compilation result to a local runtime for test purposes prior to the checking in step.

58. (New) The method of claim 56 further comprising the central compilation service making available the result of the central compilation in the runtime archive storage.

59. (New) The method of claim 56 further comprising the central compilation service storing an error result in case an error occurs during the central compilation, and making available the error result to a changer of the inactive source file causing the error.

60. (New) A local development computer configured to execute retrieving, obtaining, transferring, retrieving compilation result, locally compiling, storing, deploying and launching of the method according to claim 55.